IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	David E. Lowell	§	Art Unit:	2195
		§		
Serial No.:	10/676,922	§	Confirmation No.:	8294
		§		
Filed:	October 1, 2003	§	Examiner:	Eric Charles Wai
		§		
For:	Runtime Virtualization and Devirtualization of I/O Devices by a Virtual Machine Monitor	§ § §	Atty. Dkt. No.:	200309154-1 (HPC.0518US)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64 and 66 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company, LP. The Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 11445 Compaq Center Drive West, Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

A Notice of Appeal dated March 15, 2010, was filed in U.S. Serial No. 10/677,159. The appeal in U.S. Serial No. 10/677,159 may be related to, directly affect, or be directed affected by, or have a bearing on the Board's decision in the pending appeal.

.

III. STATUS OF THE CLAIMS

Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64 and 66 have been finally rejected and are the subject of this appeal. Claims 1, 5, 29, 31, 34, 47, 51, 55 and 65 have been cancelled.

IV. STATUS OF AMENDMENTS

No amendment after the final rejection of December 3, 2009 has been submitted. Therefore, all amendments have been entered.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element. Note also that the cited passages are provided as examples, as other passages in the specification or drawings not cited may also be relevant to the corresponding claim elements.

Independent claim 11 recites in a computer including an I/O device, a method comprising using a virtual machine monitor (Fig. 1:114) to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln. 1-7; p. 5, ¶ [0023], ln. 7-10), wherein the virtual machine monitor determines whether the I/O device is performing an I/O sequence, and delays commencing the virtualization until the virtual machine monitor determines that the I/O sequence has completed (Spec., p. 6, ¶ [0026], ln. 1-8).

Independent claim 15 recites in a computer including an I/O device, a method comprising:

using a virtual machine monitor (Fig. 1:114) to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer, wherein the I/O device has multiple modes of operations (Spec., p. 4, ¶ [0020], ln. 1-7; p. 5, ¶ [0023], ln. 7-10);

the virtual machine monitor (Fig. 1:114) determining the mode of the I/O device prior to commencing the virtualization (Spec., p. 15, ¶ [0053], ln. 2-4); and

the virtual machine monitor (Fig. 1:114) restoring the determined mode of operation after the virtualization (Spec., p. 15, ¶ [0053], ln. 6-8).

Independent claim 17 recites in a computer including hardware, a method comprising:

running a virtual machine monitor (Fig. 1:114) on the hardware (Fig. 1:110; Spec., p. 4, ¶ [0019], ln. 2-3);

running an operating system (Fig. 1:112) on the virtual machine monitor (Spec., p. 4, ¶ [0019], ln. 1-5),

wherein the hardware (Fig. 1:110) includes an I/O device, and the I/O device is already virtualized by the virtual machine monitor (Spec., p. 4, ¶ [0020], ln. 1-4; p. 5, ¶ [0023], ln. 7-10); and

devirtualizing (Fig. 6:616) the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln .4-7; p. 13, ¶ [0045], ln. 1 - ¶ [0046], ln. 8; p. 13, ¶ [0048], ln. 1 - p. 14, ¶ [0049], ln. 14).

Independent claim 35 recites a computer comprising:

hardware (Fig. 1:110) including an I/O device (Spec., p. 4, ¶ [0018], ln. 2-4); and

computer memory encoded with a virtual machine monitor (Fig. 1:114) for running on the hardware and commencing virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln. 1-7; p. 5, ¶ [0023], ln. 7-10),

wherein the virtual machine monitor (Fig. 1:114) is configured to determine whether the I/O device is performing an I/O sequence, and to delay commencing the virtualization until the virtual machine monitor determines that the I/O sequence has completed (Spec., p. 6, ¶ [0026], ln. 1-8).

Independent claim 38 recites a computer comprising:

hardware (Fig. 1:110) including an I/O device (Spec., p. 4, ¶ [0018], ln. 2-4); and

computer memory (Fig. 1:115) encoded with a virtual machine monitor (Fig. 1:114) for devirtualizing the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln. 4-7; p. 13, ¶ [0045], ln. 1 - ¶ [0046], ln. 8; p. 13, ¶ [0048], ln. 1 - p. 14, ¶ [0049], ln. 14).

Independent claim 52 recites an article for a computer including an I/O device, the article comprising computer-readable memory (Fig. 1:115) encoded with a virtual machine monitor (Fig. 1:114) for causing the computer to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln. 1-7; p. 5, ¶ [0023], ln. 7-10), wherein the virtual machine monitor determines whether the I/O device is performing an I/O sequence, the virtual machine monitor delaying the commencement of the virtualization until the virtual machine monitor determines that the I/O sequence has completed (Spec., p. 6, ¶ [0026], ln. 1-8).

Independent claim 56 recites an article for a computer including an I/O device, the article comprising computer-readable memory encoded with a virtual machine monitor (Fig. 1:114) for causing the computer to devirtualize the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer (Spec., p. 4, ¶ [0020], ln. 4-7; p. 13, ¶ [0045], ln. 1 -¶ [0046], ln. 8; p. 13, ¶ [0048], ln. 1; p. 14, ¶ [0049], ln. 8).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66 were provisionally rejected on the ground of nonstatutory obviousness-type double patenting over copending Application No. 10/677,159.

B. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66 were provisionally rejected on the ground of nonstatutory obviousness-type double patenting over copending Application No. 10/676,557.

C. Claims 8, 10, 15, 17-19, 27-28, 38, 45-46, 56-57, 64, and 66 were rejected under 35 U.S.C. § 102(e) as anticipated by Le (U.S. Patent No. 7,082,598).[1]

D. Claims 9, 20-24, 39-43, and 58-62 were rejected under 35 U.S.C. § 103(a) as unpatentable over Le (U.S. Patent No. 7,082,598).

E. Claims 25-26, 44, and 63 were rejected under 35 U.S.C. § 103(a) as unpatentable over Le in view of Nelson (U.S. Patent No. 6,961,941).

F. Claims 2-4, 6-7, 11-14, 16, 30, 32-33, 35-37, 48-50, and 52-54 were rejected under 35 U.S.C. § 103(a) as unpatentable over Nelson in view of Duvall (U.S. Patent No. 4,742,447).

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

A. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66 were provisionally rejected on the ground of nonstatutory obviousness-type double patenting over copending Application No. 10/677,159.

1. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66.

A final rejection has issued and a notice of appeal has been filed in U.S. Serial No. 10/677,159. Since it is unknown at this point whether 10/677,159 or the present application will

[1] A typographical error was made in this rejection—"56-54" should be "56-57."

issue first, and what the scope of the claims of each of the two cases will be, the nonstatutory obviousness-type double patenting rejection over claims of 10/676,577 is premature and should be withdrawn.

Reversal of the provisional nonstatutory obviousness-type double patenting of the above claims is respectfully requested.

B. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66 were provisionally rejected on the ground of nonstatutory obviousness-type double patenting over copending Application No. 10/676,557.

1. Claims 2-4, 6-28, 30, 32-33, 35-46, 48-50, 52-54, 56-64, and 66.

A final rejection has issued in U.S. Serial No. 10/676,577. Since it is unknown at this point whether 10/676,577 or the present application will issue first, and what the scope of the claims of each of the two cases will be, the nonstatutory obviousness-type double patenting rejection over claims of 10/676,577 is premature and should be withdrawn.

Reversal of the provisional nonstatutory obviousness-type double patenting rejection of the above claims is respectfully requested.

C. Claims 8, 10, 15, 17-19, 27-28, 38, 45-46, 56-57, 64, and 66 were rejected under 35 U.S.C. § 102(e) as anticipated by Le (U.S. Patent No. 7,082,598).

1. Claims 8, 15.

It is respectfully submitted that claim 15 is not anticipated by Le.

Claim 15 recites a method comprising:

> using a virtual machine monitor to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer, wherein the I/O device has multiple modes of operations;
>
> the virtual machine monitor determining the mode of the I/O device prior to commencing the virtualization; and

the virtual machine monitor restoring the determined mode of operation after the virtualization.

As purportedly disclosing a VMM to commence virtualization of the I/O device at runtime, the Examiner cited the following passages of Le: column 31, lines 39-40; column 34, lines 39-55. 12/3/2009 Office Action at 4. The cited column 34 passage of Le refers to an operation of the VMM, which intercepts requests issued by a VM. Le, 34:39-42. Also, this passage states that if the VMM needs the host OS to process an I/O request, the VMM calls an application APPx 1980, which in turn calls a driver VMX 1985. *Id.*, 34:45-48. As further explained in Le, the VMM is on the virtual side of the computer, while the host OS and system software are on the non-virtual side of the computer. *Id.*, 34:52-55; Fig. 19. There is nothing in the column 34 passage of Le cited by the Examiner that even remotely hints at a VMM **commencing** virtualization of the I/O device at runtime, where runtime is a period of execution in the computer after boot and before shutdown of the computer.

The other passage of Le cited by the Examiner (column 31, lines 39-40) refers to dynamically substituting drivers without needing to power off the system or rebooting the OS. Substituting a driver has nothing to do with the VMM commencing virtualization of the I/O device at runtime.

In arguing that Le discloses a VMM to commence virtualization of the I/O device at runtime, the Examiner stated that "a driver is installed to allow the VMM to process I/O requests, *i.e.*, the installed driver allows the device to become virtualized." 12/3/2009 Office Action at 4. Column 34 of Le does note that an agent 1850 as shown in Fig. 19 of Le is able to switch between a source driver 1842 and a target driver 1844 for a particular device 1810. Le, 34:61-64. As noted by Le, this ability by the agent 1850 to switch between drivers "makes it possible

to load a driver 1844 for the device 1810 that implements some special need of the VM/VMM without having to permanently replace the normal driver 1842." *Id.*, 34:64-67.

The agent 1850 shown in Fig. 19 of Le is on the "non-virtual" side of the computer. Thus, the agent 1850 cannot be considered the VMM. Therefore, the agent 1850 switching between drivers does not constitute the VMM commencing virtualization of the I/O device at runtime.

Claim 15 further recites that the VMM determines the mode of the I/O device prior to commencing the virtualization, and that the VMM restores the determined mode of operation after the virtualization. As purportedly disclosing the VMM determining the mode of the I/O device prior to commencing the virtualization, the Examiner cited column 34, lines 62-66, of Le This passage of Le refers to the agent 1850 switching between a source driver and a target driver, such that it is possible to load a driver for the device that implements some special need of the VM/VMM. There is nothing here that even remotely hints at the VMM determining the mode of the I/O device prior to commencing the virtualization.

The Examiner argued that it is "inherent that the VMM detects the normal driver in order to perform the switching." 12/3/2009 Office Action at 4. This statement finds no support in the teachings of Le. Moreover, this statement does not support the argument that the VMM determines the **mode** of the I/O device prior to commencing the virtualization. In fact, column 34 of Le notes that when the VMM needs the host OS to process an I/O request, it calls the application APPx 1980, and the application APPx 1980 in turn calls the driver VMX 1985. Le, 34:45-47. As noted by Le, the driver VMX 1985 forms a kind of gateway into the host OS. *Id.*, 34:48-49. Thus, from this, it is clear that the VMM interfaces into an application on the non-virtual side of Le's computer, as illustrated by the dashed arrow between the VMM and the

application APPx 1980 shown in Fig. 19 of Le. The statement in the Final Office Action that it is "inherent that the VMM detects the normal driver in order to perform the switching" is therefore wrong, since the VMM only calls the application APPx 1980. The VMM in Fig. 19 of Le would not detect a mode of any I/O device, since the VMM merely makes a call to an application. Moreover, Le makes it clear that it is the **agent 1850** (which is on the non-virtual side of the computer) that performs the switching—consequently, there would be no need for the VMM to detect the normal driver, since the agent 1850 on the non-virtual side of the computer performs the switching between the normal driver and the target driver.

A further feature of claim 15 is that the VMM restores the determined mode of operation after the virtualization. As purportedly disclosing this feature, the Examiner cited column 34, line 64 – column 34, line 11, of Le. 12/3/2009 Office Action at 5. This passage of Le refers to the agent 1850 on the non-virtual side of the computer switching between a source driver and a target driver. This passage also notes that the presence of the VM/VMM is kept transparent to the host OS and any other software components on the non-virtual side of the system. There is nothing here to even remotely hint at the VMM restoring the determined mode of operation after the virtualization, where the determined mode is the mode determined prior to commencing the virtualization.

The Examiner argued that "a switch to the normal driver can be made to restore the operation of the device for non-virtualized software components." 12/3/2009 Office Action at 5. Again, the switching between drivers is performed by the agent 1850 that is on the non-virtual side of the computer. Thus, the agent 1850 switching between drivers has nothing to do with the **VMM** restoring the determined mode of operation after virtualization.

In view of the various errors made in the rejection of claim 15 over Le, it is clear that claim 15 and its dependent claims are allowable over Le.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claim 10.

Claim 10 depends from claim 15 and is therefore allowable for at least the same reasons as claim 15. Moreover, claim 10 further recites that the VMM commences the virtualization **during an I/O sequence**. As purportedly disclosing this feature of claim 10, the Examiner cited column 31, lines 48-55, of Le. 12/3/2009 Office Action at 5. The cited passage of Le states that a standard driver can be reinstated just as easily as it was replaced. This passage notes that driver substitution can be performed on a temporary basis. As discussed above, the switching of drivers is accomplished by an agent 1850 in the non-virtual side of the computer, which is clearly not the VMM. Moreover, it is noted that switching between drivers has nothing to do with a VMM commencing virtualization **during an I/O sequence**. Claim 10 is therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

3. Claims 17, 18, 28.

Independent claim 17 recites a method in which the I/O device is already virtualized by the VMM, and that the I/O device can be devirtualized at runtime, where runtime is a period of execution in the computer after boot and before shutdown of the computer.

With respect to the "devirtualizing" clause of claim 17, the Examiner cited the following passages of Le: column 34, line 62 – column 35, line 11. 12/3/2009 Office Action at 5. The Examiner argued that "the agent can switch between drivers for the device (*i.e.*, devirtualizing

the device) so that the 'non-virtualized' side of the system can still perform I/O requests." *Id.* at 5-6.

The switching between drivers is performed by an agent on the non-virtual side of the computer in Fig. 19 of Le. There is no hint in Le that switching between drivers constitutes **devirtualizing** an I/O device—it merely substitutes one driver for another driver. In addition, Le provides no hint that the VMM performs any devirtualization of an I/O device at runtime. Even though the agent 1850 on the non-virtual side of the computer has switched between drivers, the VMM shown in Fig. 19 of Le still continues to perform virtualization by intercepting requests from the VM of Fig. 19. Thus, the switching of drivers does not stop virtualization—the switching of drivers is directed to a completely unrelated concept.

The Examiner also cited column 31, lines 39-55, of Le, which refers to driver substitution. 12/3/2009 Office Action at 6. Again, driver substitution has nothing to do with a VMM devirtualizing the I/O device at runtime. Switching drivers does not stop virtualization, since the VMM would still continue to intercept requests by the VM even after drivers have been substituted.

In view of the foregoing, it is clear that the rejection of claim 17 and its dependent claims over Le is erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

4. Claim 19.

Claim 19 depends from claim 17 and is therefore allowable for at least the same reasons as claim 17. Moreover, claim 19 further states that devirtualization includes stopping I/O device emulation at runtime. With respect to claim 19, the Examiner cited column 31, lines 39-55, of Le. 12/3/2009 Office Action at 6. Switching between drivers does not effect the emulation or

virtualization as performed by the VM. It is clear that even after the drivers have been switched, the VMM would still continue to virtualize an I/O device by intercepting requests issued by the VM.

Thus, it is clear that Le provides absolutely no hint of the subject matter of claim 19.

Reversal of the final rejection of the above claim is respectfully requested.

5. Claim 27.

Claim 27 depends from claim 17 and is therefore allowable for at least the same reasons as claim 17. Moreover, claim 27 further recites:

> the virtual machine monitor determining the mode of the I/O device prior to commencing the devirtualization; and
>
> the virtual machine monitor restoring the determined mode of operation after devirtualization.

Tasks similar to these tasks are also recited in claim 15, discussed above. For reasons similar to those stated above with respect to claim 15 with respect to these elements, claim 27 is further allowable over Le.

Reversal of the final rejection of the above claim is respectfully requested.

6. Claims 38, 46, 56, 66.

Independent claim 38 is allowable over Le for reasons similar to those stated above with respect to claim 17, since Le fails to disclose a VMM devirtualizing an I/O device at runtime, where runtime is a period of execution in the computer after boot and before shutdown of the computer.

Claim 38 and its dependent claims are therefore clearly allowable over Le.

Claim 56 and its dependent claims are allowable for similar reasons.

Reversal of the final rejection of the above claims is respectfully requested.

7. Claim 45.

Claim 45 depends from claim 38 and is therefore allowable for at least the same reasons as claim 38. Moreover, claim 45 is further allowable for similar reasons as for claim 27.

Reversal of the final rejection of the above claim is respectfully requested.

8. Claim 57.

Claim 57 depends from claim 56 and is therefore allowable for at least the same reasons as claim 56. Moreover, claim 57 further recites that the devirtualization includes seizing emulation of the I/O device at runtime. Claim 57 is therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

9. Claim 64.

Claim 64 depends from claim 56 and is therefore allowable for at least the same reasons as claim 56. Moreover, claim 64 is further allowable for similar reasons as stated above with respect to claim 27.

Reversal of the final rejection of the above claim is respectfully requested.

D. Claims 9, 20-24, 39-43, and 58-62 were rejected under 35 U.S.C. § 103(a) as unpatentable over Le (U.S. Patent No. 7,082,598).

1. Claims 9, 24, 43, 61.

In view of the allowability of base claims over Le, the obviousness rejection of the foregoing dependent claims over Le has been overcome.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claims 20-22, 39-41, 58-60.

In view of the allowability of base claims over Le, the obviousness rejection of the foregoing dependent claims over Le has been overcome. Moreover, claim 20 further recites that the VMM emulates the I/O device prior to devirtualization, and that the devirtualization includes allowing the VMM to temporarily stop the OS from commencing a new I/O sequence. The Examiner conceded that Le fails to disclose the foregoing claimed features.

However, the Examiner simply stated that it would have been obvious to provide such features. 12/3/2009 Office Action at 8. The Examiner has not provided any rationale regarding why the features of claim 20 conceded to be missing from Le would have been obvious. The obviousness rejection is therefore defective since the Examiner has failed to establish a *prima facie* case of obviousness.

It is clear that Le provides absolutely no hint whatsoever of the VMM temporarily stopping the OS from commencing a new I/O sequence. In fact, the passages of Le relied upon by the Examiner in rejecting base claims refers to an agent on the non-virtual side of the computer system switching between different drivers. However, even if the agent switches between different drivers, the VMM continues to perform virtualization. The VMM in Le does nothing to temporarily stop the OS from commencing a new I/O sequence.

Claim 20 and its dependent claims are therefore further allowable for the foregoing reasons.

Claims 39-41, and 58-60 are further allowable for similar reasons.

Reversal of the final rejection of the above claims is respectfully requested.

3. Claims 23, 42, 62.

In view of the allowability of base claims over Le, the obviousness rejection of dependent claims 23, 42, and 62 have also been overcome.

Moreover, claim 23 further recites:

> the virtual machine monitor logging I/O accesses by the operating system to the I/O device during devirtualization, and
>
> replaying the log to the I/O device after devirtualization, wherein the I/O accesses by the operating system are deferred during the devirtualization of the I/O device.

The Examiner conceded that Le fails to disclose the subject matter of claim 23, but then makes a conclusory statement that it would have been obvious to include the features of claim 23. 12/3/2009 Office Action at 9. The Examiner did not provide any rationale to support this conclusory statement. The obviousness rejection is therefore defective since the Examiner has failed to establish a *prima facie* case of obviousness.

Nothing in Le even remotely hints at a VMM logging I/O accesses by the OS to the I/O device during devirtualization, and then replaying the log to the I/O device after devirtualization, where the I/O accesses by the OS are deferred during the devirtualization of the I/O device.

Claim 23 (and also claims 42 and 62) are therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claims is respectfully requested.

E. Claims 25-26, 44, and 63 were rejected under 35 U.S.C. § 103(a) as unpatentable over Le in view of Nelson (U.S. Patent No. 6,961,941).

1. Claims 25, 26, 44, 63.

In view of the allowability of base claims over Le, the obviousness rejection of dependent claims 25, 26, 44, and 63 over Le and Nelson has been overcome.

Reversal of the final rejection of the above claims is respectfully requested.

F. Claims 2-4, 6-7, 11-14, 16, 30, 32-33, 35-37, 48-50, and 52-54 were rejected under 35 U.S.C. § 103(a) as unpatentable over Nelson in view of Duvall (U.S. Patent No. 4,742,447).

1. Claims 2-4, 6, 7, 11-13, 16, 30, 32, 33, 35, 37, 48-50, 52, 54.

Independent claim 11 was rejected as purportedly obvious over Nelson in view of Duvall. It is respectfully submitted that the obviousness rejection of claim 11 is erroneous.

To make a determination under 35 U.S.C. § 103, several basic factual inquiries must be performed, including determining the scope and content of the prior art, and ascertaining the differences between the prior art and the claims at issue. *Graham v. John Deere Co.,* 383 U.S. 1, 17, 148 U.S.P.Q. 459 (1965). Moreover, as held by the U.S. Supreme Court, it is important to identify a reason that would have prompted a person of ordinary skill in the art to combine reference teachings in the manner that the claimed invention does. *KSR International Co. v. Teleflex, Inc.,* 127 S. Ct. 1727, 1741, 82 U.S.P.Q.2d 1385 (2007).

In the rejection of claim 11, the Examiner conceded that Nelson fails to disclose that the VMM determines whether the I/O device is performing an I/O sequence, and delaying commencing the virtualization until the VMM determines that the I/O sequence has completed. 12/3/2009 Office Action at 12. However, the Examiner cited Duvall, and more specifically, to column 12, lines 29-34, of Duvall as purportedly disclosing the foregoing claimed subject matter. The cited column 12 passage of Duvall refers to page fault processing. As explained in this passage of Duvall, synchronous page fault processing is a traditional type of page fault processing, where the faulting process is forced to wait until the I/O required to resolve the page fault is complete. However, waiting for completion of the I/O required to resolve the page fault to complete, before performing page fault processing, is completely unrelated to a virtual

machine monitor delaying commencing of the virtualization of the I/O device until the virtual machine monitor determines that the I/O sequence has completed.

Therefore, it is clear that Duvall does not provide any teaching or hint of claimed subject matter that is clearly missing from Nelson. Thus, the hypothetical combination of Nelson and Duvall would not have led to the claimed subject matter.

Moreover, a person of ordinary skill in the art would have found no reason whatsoever to incorporate the page fault processing technique of Duvall into the virtual machine monitoring environment of Nelson. There is absolutely no hint whatsoever of page fault handling in the teachings of Nelson. Thus, no reason existed to combine the teachings of Nelson and Duvall to achieve the claimed subject matter.

The obviousness rejection of claim 11 and its dependent claims are therefore clearly erroneous.

Independent claims 35 and 52 (and their respective dependent claims) are similarly allowable over Nelson and Duvall.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claims 14, 36, 53.

Claims 14, 36, and 53 depend from respective base claims 11, 35, and 52, and are therefore allowable for at least the same reasons as corresponding base claims. Moreover, claim 14 further recites that the VMM temporarily pauses an I/O sequence by emulating the I/O device as being busy. With respect to claim 14, the Examiner cited column 3, lines 41-43, of Nelson. 12/3/2009 Office Action at 14. This passage of Nelson states that handling of interrupts are forwarded to a first operating system (COS), and interrupts that are generated by host-managed devices are preferably delayed until a subsequent instance of running of the COS. However,

there is nothing in this passage of Nelson, or in any other passages of Nelson, regarding a virtual machine monitor temporarily pausing an I/O sequence by emulating the I/O device as being busy. Duvall also fails to provide any hint of the foregoing subject matter.

Therefore, claim 14 (and claims 36 and 53) are further allowable for the foregoing reasons.

Reversal of the final rejection of the above claims is respectfully requested.

CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: April 26, 2010

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

VIII. APPENDIX OF APPEALED CLAIMS

The claims on appeal are (claims 1, 5, 29, 31, 34, 47, 51, 55 and 65 have been cancelled):

2. The method of claim 11, wherein the computer further includes a CPU, wherein the virtual machine monitor is in control of the CPU prior to the runtime virtualization of the I/O device.

3. The method of claim 11, wherein the virtualization is performed transparently to an operating system.

4. The method of claim 11, wherein the I/O device is compatible with the virtualized I/O device.

6. The method of claim 11, further comprising configuring hardware to trap I/O accesses, and enabling the virtual machine monitor to emulate the I/O device in response to the trapped I/O accesses.

7. The method of claim 6, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

8. The method of claim 15, wherein the virtual machine monitor commences the virtualization between I/O sequences.

9. The method of claim 8, wherein the virtual machine monitor commences the virtualization by intercepting I/O accesses; wherein the virtual machine monitor uses the intercepted I/O accesses to update a state machine, whereby the state machine reflects a state of the I/O device; and wherein the virtual machine monitor examines transitions in the state machine to determine whether the I/O device is in the middle of an I/O sequence.

10. The method of claim 15, further comprising the virtual machine monitor commencing the virtualization during an I/O sequence.

11. In a computer including an I/O device, a method comprising using a virtual machine monitor to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer, wherein the virtual machine monitor determines whether the I/O device is performing an I/O sequence, and delays commencing the virtualization until the virtual machine monitor determines that the I/O sequence has completed.

12. The method of claim 11, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

13. The method of claim 11, wherein I/O device interrupts are directed to an operating system prior to the runtime virtualization of the I/O device; and wherein the I/O device interrupts are directed to the virtual machine monitor during and after the virtualization of the I/O device.

14. The method of claim 11, wherein the virtual machine monitor temporarily pauses an I/O sequence by emulating the I/O device as being busy.

15. In a computer including an I/O device, a method comprising:

using a virtual machine monitor to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer, wherein the I/O device has multiple modes of operations;

the virtual machine monitor determining the mode of the I/O device prior to commencing the virtualization; and

the virtual machine monitor restoring the determined mode of operation after the virtualization.

16. The method of claim 11, further comprising devirtualizing the I/O device at runtime following the runtime virtualization.

17. In a computer including hardware, a method comprising:
running a virtual machine monitor on the hardware;
running an operating system on the virtual machine monitor,
wherein the hardware includes an I/O device, and the I/O device is already virtualized by the virtual machine monitor; and
devirtualizing the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer.

18. The method of claim 17, wherein the devirtualization is performed transparently to the operating system.

19. The method of claim 17, wherein the devirtualization includes stopping I/O device emulation at runtime.

20. The method of claim 17, wherein the virtual machine monitor emulates the I/O device prior to devirtualization; and wherein the devirtualization includes allowing the virtual machine monitor to temporarily stop the operating system from commencing a new I/O sequence.

21. The method of claim 20, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

22. The method of claim 20, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

23. The method of claim 20, further comprising:

the virtual machine monitor logging I/O accesses by the operating system to the I/O device during devirtualization, and

replaying the log to the I/O device after devirtualization, wherein the I/O accesses by the operating system are deferred during the devirtualization of the I/O device.

24. The method of claim 17, wherein the virtual machine monitor waits for I/Os initiated by the virtual machine monitor's driver for the I/O device to complete, and for all expected interrupts from the device to arrive, before ceasing device emulation.

25. The method of claim 17, further comprising re-directing interrupts from interrupt handlers in the virtual machine monitor to interrupt handlers in the operating system after performing the devirtualizing.

26. The method of claim 17, further comprising, after performing the devirtualizing, configuring the hardware so accesses by the operating system to the I/O device no longer trap to the virtual machine monitor.

27. The method of claim 17, wherein the I/O device has multiple modes of operations, the method further comprising:

the virtual machine monitor determining the mode of the I/O device prior to commencing the devirtualization; and

the virtual machine monitor restoring the determined mode of operation after devirtualization.

28. The method of claim 17, further comprising virtualizing the I/O device at runtime again after performing the devirtualizing at runtime.

30. The computer of claim 35, wherein the I/O device is compatible with the virtualized I/O device.

32. The computer of claim 35, wherein the hardware is configured to trap I/O accesses, and the virtual machine monitor is enabled to emulate the I/O device in response to the trapped I/O accesses.

33. The computer of claim 32, wherein the virtual machine monitor is configured to use memory management to trap the I/O accesses.

35. A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine monitor for running on the hardware and commencing virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer,

wherein the virtual machine monitor is configured to determine whether the I/O device is performing an I/O sequence, and to delay commencing the virtualization until the virtual machine monitor determines that the I/O sequence has completed.

36. The computer of claim 35, wherein the virtual machine monitor is configured to temporarily pause the I/O sequence by emulating the I/O device as being busy.

37. The computer of claim 35, wherein the runtime virtualization includes using the virtual machine monitor to emulate I/O device interrupts.

38. A computer comprising:

hardware including an I/O device; and

computer memory encoded with a virtual machine monitor for devirtualizing the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer.

39. The computer of claim 38, wherein the virtual machine monitor is configured to emulate the I/O device prior to commencing the devirtualization; and wherein the virtual machine is configured to commence the devirtualization by temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

40. The computer of claim 39, wherein the virtual machine monitor is configured to temporarily stop the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

41. The computer of claim 39, wherein the virtual machine monitor is configured to bound the amount of time the operating system processing is temporarily stopped.

42. The computer of claim 38, wherein the virtual machine monitor is configured to log I/O accesses by an operating system to the I/O device during devirtualization, and to replay the log to the I/O device after devirtualization.

43. The computer of claim 39, wherein the virtual machine monitor is configured to wait for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

44. The computer of claim 38, wherein the hardware is configured so operating system accesses to the I/O device no longer trap to the virtual machine monitor after the devirtualization.

45. The computer of claim 38, wherein the I/O device has multiple modes of operations; wherein the virtual machine monitor is configured to determine the mode of the I/O device prior to commencing the devirtualization; and wherein the virtual machine monitor is configured to restore the determined mode of operation after the I/O device has been devirtualized.

46. The computer of claim 38, wherein the virtual machine monitor is configured to further virtualize the I/O device after having devirtualized the I/O device at runtime.

48. The article of claim 52, wherein the virtualization includes commencing I/O device emulation at runtime.

49. The article of claim 48, wherein the virtual machine monitor configures the hardware to trap I/O accesses, and enables the virtual machine monitor to emulate the I/O device in response to the trapped I/O devices.

50. The article of claim 49, wherein the virtual machine monitor uses memory management to trap the I/O accesses.

52. An article for a computer including an I/O device, the article comprising computer-readable memory encoded with a virtual machine monitor for causing the computer to commence virtualization of the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer, wherein the virtual machine monitor determines whether the I/O device is performing an I/O sequence, the virtual machine monitor delaying the commencement of the virtualization until the virtual machine monitor determines that the I/O sequence has completed.

53. The article of claim 52, wherein the virtual machine monitor temporarily pauses the I/O sequence by emulating the I/O device as being busy.

54. The article of claim 52, wherein the virtual machine monitor emulates I/O device interrupts during the runtime virtualization.

56. An article for a computer including an I/O device, the article comprising computer-readable memory encoded with a virtual machine monitor for causing the computer to devirtualize the I/O device at runtime, wherein runtime is a period of execution in the computer after boot and before shutdown of the computer.

57. The article of claim 56, wherein the devirtualization includes ceasing emulation of the I/O device at runtime.

58. The article of claim 57, wherein the devirtualization includes temporarily stopping an operating system running on the virtual machine monitor from commencing a new I/O sequence.

59. The article of claim 58, wherein the virtual machine monitor temporarily stops the operating system by emulating the I/O device as being in a "busy" or "device not ready" state.

60. The article of claim 58, wherein the virtual machine monitor bounds the amount of time the operating system processing is temporarily stopped.

61. The article of claim 57, wherein the virtual machine monitor waits for I/Os initiated by a virtual machine monitor driver for the I/O device to complete, and for all expected interrupts from the I/O device to arrive, before ceasing device emulation.

62. The article of claim 56, wherein the virtual machine monitor logs I/O accesses by an operating system to the I/O device during devirtualization, and replays the log to the I/O device after devirtualization.

63. The article of claim 56, wherein the virtual machine monitor, configures the hardware so operating system accesses to the I/O device do not trap to the virtual machine monitor.

64. The article of claim 56, wherein the I/O device has multiple modes of operations; and wherein the virtual machine monitor determines the mode of the I/O device prior to commencing devirtualization; and restore the determined mode of operation after the I/O device has been devirtualized.

66. The article of claim 56, wherein the virtual machine monitor causes the computer to further virtualize the I/O device after having devirtualized the I/O device at runtime.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

A Notice of Appeal dated March 15, 2010, was filed in U.S. Serial No. 10/677,159. No decision on appeal has been rendered in this case.